

The background of the slide is a stylized landscape. It features a light blue sky with three white, fluffy clouds. In the top right corner, there is a bright yellow sun with a thick orange border. The ground is represented by green rolling hills. On the left hill, there is a single green tree with a brown trunk and several small pink flowers at its base. On the right hill, there are two green trees with brown trunks and more small pink flowers scattered around them.

Game Engines from Scratch

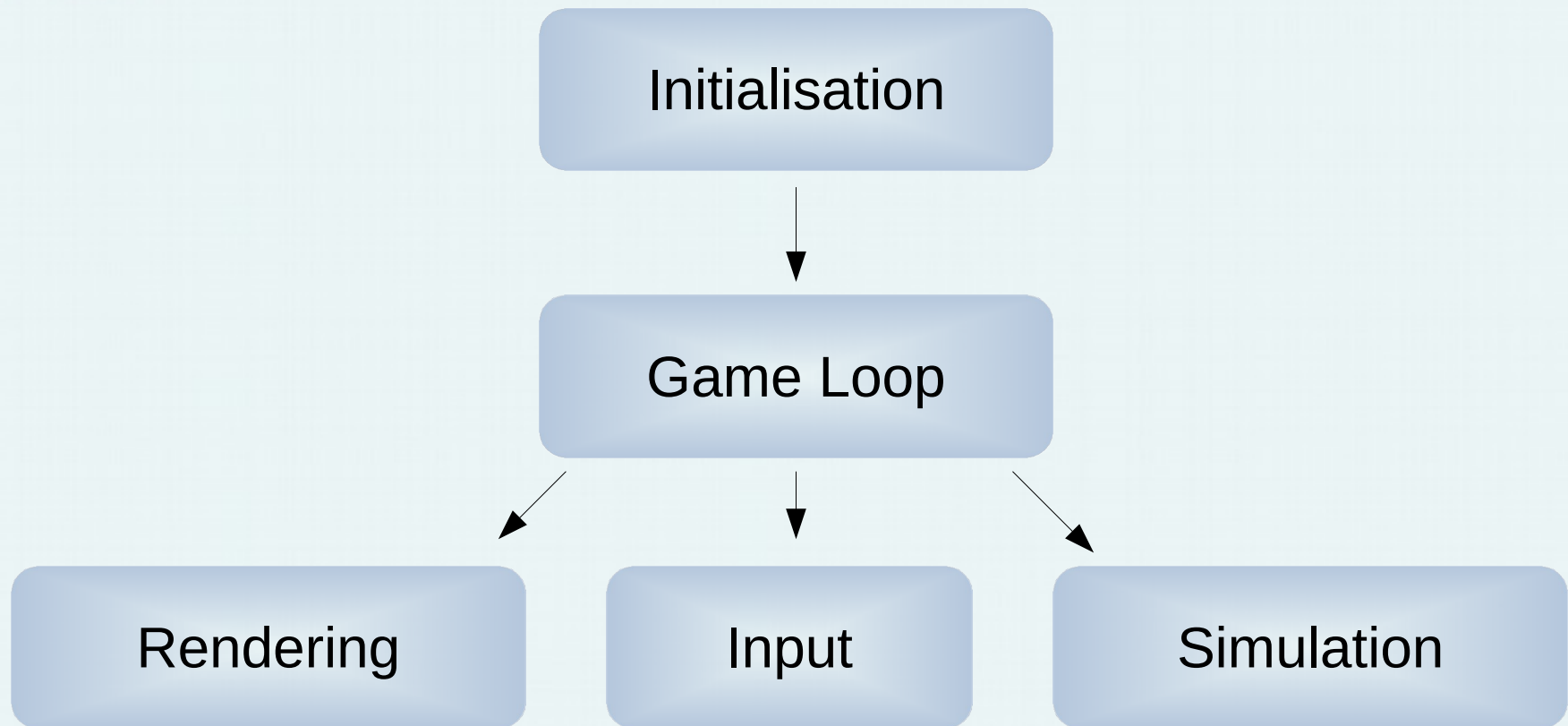
Samuel Davidson

Why DIY?

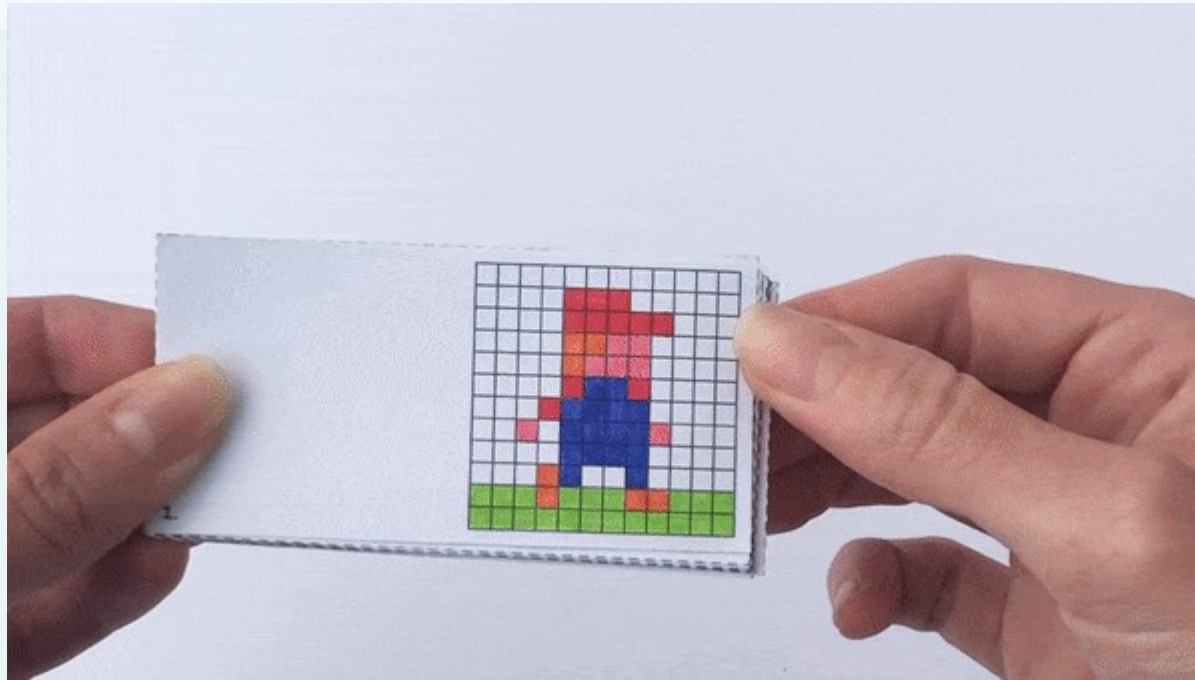
- You were dropped as a baby
- You want to gain XP
- Portfolio piece (non/technical bridge)
- **You want to build a good game**
- Your game has unique requirements
(UE5/Unity/Swarm Engine)



General Architecture



Rendering/Simulation Decouple



Rendering
- renders scene
- governs FPS (deltaTime)

Simulation
- runs whole sim
- completes each frame

Architecture Walkthrough

- Sand Sim Engine
- Mr Blue Square



OOP & DOD

Object Oriented Programming

Class Car {

int ID; // 4 bytes

- Color Colour; // 8 bytes
- byte Wheels; // 1 byte
- float BrakeForce; // 2 bytes
- float Velocity; // 8 bytes
- bool IsBraking; // 1 bit

}

Function ApplyBrakes(Car car) {

if (car.IsBraking) {

car.Velocity -= car.BrakeForce;

}

}

Array<Car> Cars = [Car1, Car2, Car3, ...];

for (int i = 0; i < Cars.length; i++) {

ApplyBrakes(Cars[i]);

}

Data Oriented Design

Struct Cars{

int[] IDs; // 4 bytes each

Color[] Colours; // 8 bytes each

byte[] Wheels; // 1 byte each

float[] BrakeForces; // 2 bytes each

float[] Velocities; // 8 bytes each

bool[] IsBraking; // 1 bit each

}

Function ApplyBrakes(float[] velocities, float[] brakeForces, bool[] isBraking) {

for (int i = 0; i < velocities.length; i++) {

if (isBraking[i]) {

velocities[i] -= brakeForces[i];

}

}

}

Cars cars = {

IDs: [0, 1, 2, 3],

IsBraking: [true, true, false, true],

Colours: [Red, Blue, Green, Yellow],

Wheels: [4, 4, 3, 4],

BrakeForces: [2.0, 1.6, 2.2, 1.8],

Velocities: [70.0, 50.0, 60.0, 80.0]

};

BreakingCars = [0,1,3];

ApplyBrakes(BreakingCars.Velocities[id], BreakingCars.BrakeForces[id]);



OOP & DOD In Memory

Object

Size =	23.1b
Useful Size =	10.1b
Used Size =	23.1b
Wasted =	13b

Entity

Size =	23.1b
Useful Size =	10.1b
Used Size =	10.1b
Wasted =	0



What's Next?

- Wave Function Collapse
- Gamified Neural Net Interface



Thank you!

Github



Youtube



Samuel Davidson

